Jazida: um sistema de indexação e busca em cluster para bibliotecas digitais

Aécio Solano R. Santos
aecio@ifpi.edu.br

Valéria Oliveira Costa
valeria@ifpi.edu.br

Lidijanne de M. Santos
lidijanne@ifpi.edu.br

Instituto Federal do Piauí - Laboratório de Pesquisa em Sistemas de Informação

ABSTRACT

This paper describes the architecture of a text and image retrieval distributed system for the EPCT digital library. The system was designed to run on commodity hardware clusters, solving scalability and high availability problems of the current version of the system. It shows also some technologies that will be used. Finally, some preliminary tests demonstrate the efficiency of the indexing architecture.

Categories and Subject Descriptors

H.3.4 [**Information Storage And Retrieval**]: Systems and Software—*Distributed Systems*

General Terms

Design, Reliability, Performance

Keywords

digital libraries, distributed indexing, distributed search engine, lucene

1. INTRODUÇÃO

A Biblioteca Digital da Rede Federal de Educação Profissional, Científica e Tecnológica (EPCT)[1] é um projeto desenvolvido colaborativamente entre o Instituto Federal do Piauí (IFPI) e o Instituto Federal Fluminense (IFF). O projeto visa a disseminação de material científico e tecnológico produzido na rede de instituições de EPCT.

A Biblioteca Digital é desenvolvida utilizando Arquitetura Orientada a Serviços. O Laboratório de Pesquisa em Sistemas de Informação (LAPESI) do IFPI é responsável por desenvolver o serviço de indexação e busca por conteúdo e metadados de textos e imagens. A versão atual está preparada para funcionar em apenas um computador, que fica responsável por responder a todas as requisições ao serviço.

[1]http://www.renapi.org/biblioteca-digital

Este trabalho propõe uma arquitetura distribuída para o serviço de indexação e busca, com a utilização de clusters computacionais visando aumentar a escalabilidade e a disponibilidade do serviço.

O restante deste trabalho está estruturado da seguinte forma: a seção 2 descreve a versão atual do sistema, a seção 3 mostra alguns trabalhos relacionados e a seção 4 as principais tecnologias que serão utilizadas para implementação do projeto. Finalmente, a seção 5 descreve a arquitetura do Jazida e a seção 6 mostra os resultados de implementações iniciais. As conclusões e trabalhos futuros estão na seção 7.

2. SERVIÇO DE INDEXAÇÃO E BUSCA

A versão atual do serviço de indexação e busca da biblioteca digital pode ser acessada utilizando o protocolo XML-RPC, uma tecnologia para comunicação entre sistemas baseada no envio de arquivos XML, como ilustrado na Figura 1.

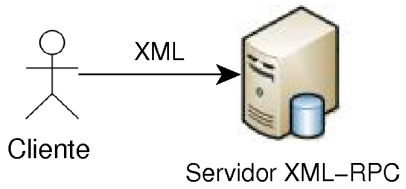

Figure 1: Arquitetura da versão atual

Na operação de indexação, por exemplo, o servidor recebe um arquivo XML contendo metadados do documento a ser indexado e o conteúdo do mesmo, interpreta-o e realiza a indexação. No núcleo do servidor XML-RPC está o *Opala*, uma biblioteca desenvolvida no LAPESI baseada no Lucene[2] e no LIRe[3] (Lucene Image Retrieval)[3]. O Opala é responsável por realizar as operações de indexação e busca do serviço. Atualmente o servidor responde pelos 8 métodos descritos na Tabela 1.

3. TRABALHOS RELACIONADOS

Foram estudados alguns projetos que permitem buscas em clusters. Porém, nenhum deles atende a necessidade da biblioteca digital da EPCT de realizar busca de imagem por conteúdo.

[2]http://lucene.apache.org
[3]http://semanticmetadata.net/lire

Table 1: Métodos disponíveis

MÉTODO	DESCRIÇÃO
TextIndexer.addText	Indexa um documento
TextIndexer.delText	Remove um documento do índice
TextIndexer.updateText	Atualiza um documento do índice
TextSearcher.searchText	Realiza busca por conteúdo no índice
ImageIndexer.addImage	Adiciona uma imagem no índice
ImageIndexer.delImage	Remove uma imagem do índice
ImageIndexer.updateImage	Atualiza uma imagem do índice
ImageSearcher.searchImage	Realiza busca de imagens por conteúdo no índice

3.1 Distributed Lucene

Distributed Lucene[1] é um sistema de indexação de texto desenvolvido na HP Laboratories, baseado em dois projetos open-source da Apache: Lucene e Hadoop. Permite indexação, atualização e remoção de documentos do índice. A arquitetura do Distributed Lucene está descrita em [1], mas o sistema não foi disponibilizado.

3.2 Katta

Katta[4] é um sistema de busca distribuído escalável e tolerante a falhas. Provê busca de documentos de texto em índices distribuídos e replicados para suportar grandes conjuntos de dados e altas cargas. Índices do Katta devem ser construídos externamente e depois instalados. Utiliza o Zookeeper[5] para coordenação dos nós do cluster. Ao contrário do Distributed Lucene, não permite atualização do índice.

3.3 Elastic Search

Elastic Search[6] é um sistema de busca distribuído escalável para computação em nuvem que permite indexação, busca e deleção de documentos de texto. Pode ser acessado por troca de arquivos JSON (JavaScript Object Notation) via REST (Representational State Transfer), ou através de APIs (Application Programming Interface) disponibilizadas para as linguagens Java e Groovy.

4. TECNOLOGIAS UTILIZADAS

As principais tecnologias escolhidas para implementação do Jazida estão descritas a seguir.

4.1 Lucene

Lucene é uma biblioteca de recuperação de informação de alta performance. Para realizar buscas, primeiro os documentos devem passar por um processo chamado"indexação", em que o texto é processado por analisadores. Em seguida é construída uma estrutura de dados chamada índice invertido, que é utilizada durante as buscas.

[4]http://katta.sourceforge.net
[5]Mais detalhes na seção 4.3
[6]http://www.elasticsearch.com

4.2 Hadoop

Hadoop[7] é um projeto da Apache que desenvolve software para computação distribuída escalável e confiável. Foi desenvolvido com base em trabalhos da Google para processamento de dados em larga escala. Hadoop conta com vários subprojetos como o Hadoop MapReduce, um framework para processamento de dados em larga escala; o HDFS (Hadoop Distributed File System) um sistema de arquivos distribuído; o HBase um banco de dados orientado a colunas, entre outros. Neste trabalho será utilizada somente a camada de comunicação inter-processos e RPC do Hadoop (Hadoop IPC/RPC).

4.3 Zookeeper

O Zookeeper é o serviço de coordenação distribuída do Hadoop. Provê várias ferramentas para o desenvolvimento de aplicações distribuídas que podem se recuperar de falhas parciais. O Zookeeper permite armazenar pequenas quantidades de dados organizadas hierarquicamente como em um sistema de arquivos. Dessa forma é possível implementar soluções para uma grande diversidade de problemas, como sincronização de processos distribuídos, entre outros. Em avaliações de performance feitas pela Yahoo![8], o Zookeeper foi capaz de realizar 10.000 operações por segundo em cargas de trabalho com predominância de escrita[4]. Para cargas com predominância de leitura, a performance foi muito melhor.

5. ARQUITETURA DO JAZIDA

O Jazida é um sistema distribuído projetado para funcionar em clusters de servidores comuns. Sua arquitetura foi desenvolvida baseada no estudo de outros sistemas distribuídos como o Katta, o Google File System [2] e o HDFS. O Jazida é constituído de vários componentes como: *Master, DataNode, JazidaClient, Zookeeper Service*. A Figura 2 mostra os componentes do Jazida.

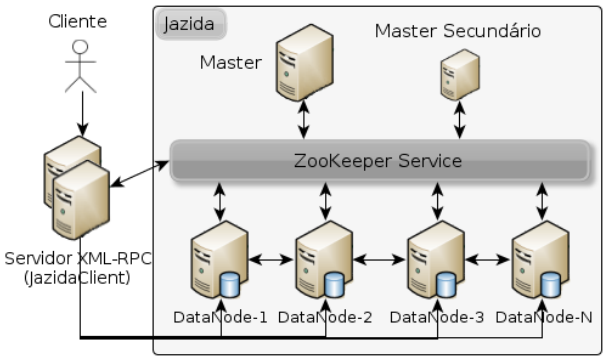

Figure 2: Componentes e arquitura do Jazida

Um DataNode é responsável por armazenar índices e servir chamadas remotas equivalentes aos métodos descritos na Tabela 1. Deve ainda armazenar réplicas de índices de outros DataNodes. Durante sua inicialização, o DataNode publica-se no Zookeeper Service como disponível. Caso ocorra uma falha no DataNode, o Zookeeper Service se encarrega de

[7]http://hadoop.apache.org
[8]http://www.yahoo.com

removê-lo da lista de DataNodes disponíveis e notifica o Master da falha ocorrida.

O Master é responsável por gerenciar falhas de DataNodes e réplicas do índice. Ao iniciar, o Master se inscreve no Zookeeper Service para receber notificações de falhas de DataNodes. Além disso, publica-se no Zookeeper como mestre, assim um Master secundário poderá monitorar ocorrência de falhas no Master e assumir seu lugar, quando necessário.

Para uma aplicação se comunicar com o cluster, deve utilizar a biblioteca JazidaClient. Esta biblioteca esconde da aplicação a existência de vários índices distribuídos nos DataNodes. Para realizar uma indexação, a biblioteca primeiro se comunica com o Zookeeper Service para descobrir os DataNodes ativos no cluster e então escolhe um DataNode para indexar o documento. A biblioteca deve implementar uma política de balanceamento de carga para escolha do DataNode, como o algoritmo Round Robin (escolhe nós de uma forma circular) ou um algoritmo que leve em consideração a carga atual de cada DataNode. A comunicação entre a biblioteca JazidaClient e cada DataNode deve ser feita através de RPC (Remote Procedure Call). Para este projeto foi escolhido o Hadoop IPC/RPC pela simplicidade e controle de como os objetos são serializados antes de serem envidados pela rede.

Para fazer uma busca, a biblioteca deverá descobrir os nós ativos, assim como na indexação, e depois enviar uma chamada de busca em paralelo para cada DataNode. Em seguida os resultados de cada DataNode deverão ser mesclados em um e ordenados de acordo com a relevância. As operações de atualização e deleção devem ocorrer de forma análoga as já descritas.

O servidor de XML-RPC que atualmente tem o Opala em seu núcleo, agora deverá utilizar a biblioteca JazidaClient para acessar o cluster. O Opala será utilizado em cada DataNode para realizar as indexações que chegam por RPC.

6. RESULTADOS PRELIMINARES

Foi realizada uma implementação inicial do modelo de indexação paralela para analisar a escalabilidade do método.

Para os experimentos foram utilizados computadores com 2Gb de memória, processador AMD Athlon II X2 250 com o sistema operacional Ubuntu 10.04 LTS. Os computadores foram interligados utilizando uma rede Gigabit Ethernet com cabos de rede CAT5.

Na experiênica, um processo lê arquivos de texto do disco rígido local e envia para DataNodes distribuídos na rede de forma paralela utilizando o mecanismo de IPC/RPC do Hadoop. Cada computador recebe o documento e indexa em um índice do lucene armazenado no sistema de arquivos local. Neste experimento foram indexados 10.000 arquivos de texto totalizando 49,1 MB.

Para a comparação de um sistema sequencial com uma versão paralela, é natural utilizar o aumento de velocidade. A figura 3 mostra um gráfico de escalabilidade levando em consideração o tempo total de indexação dos arquivos de texto em relação com a quantidade de máquinas utilizadas para realização do trabalho. O tempo esperado foi calculado baseado no tempo de indexação em apenas uma máquina (T_1) e no número de máquinas (n) utilizadas para realizar a indexação. Para o calculo foi utilizada a fórmula $T_n = \frac{T_1}{n}$, sendo que T_n indica o tempo esperado para n máquinas realizarem a indexação. Os tempos obtidos no experimento foram todos melhores do que o esperado. Isto pode ser explicado pela distribuição do índice em vários índices menores. Foi constatado em testes, que a medida que a quantidade de documentos de um índice aumenta a performance de indexação de um novo documento diminui. Então manter vários índices com menos documentos possibilitou uma indexação mais rápida.

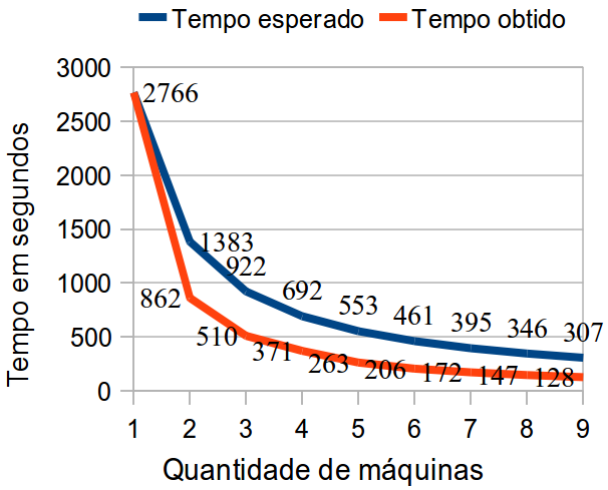

Figure 3: Resultados obtidos

7. CONCLUSÃO

Este trabalho propôs uma arquitetura para indexação e busca de documentos e imagens em clusters de servidores comuns (*commodity hardware*) reutilizando a arquitetura já existente. Experimentos realizados demostraram a eficiência da abordagem utilizada para escalabilidade na indexação de documentos, porém, a arquitetura proposta ainda deve ser implementada, bem como, a realização de análises de escalabilidade mais detalhadas incluindo análise de tráfego de rede, utilização de CPU e memória. Ainda deve ser analisada também a escalabilidade na busca, visto que é outro ponto crítico do sistema.

8. AGRADECIMENTOS

Agradecemos à Secretaria de Educação Profissional e Tecnológica (SETEC/MEC) pelo financiamento do projeto.

9. REFERENCES

[1] M. H. Butler and J. Rutherford. Distributed lucene: A distributed free text index for hadoop. Technical report, HP Laboratories, 2008.

[2] S. Ghemawat, H. Gobioff, and S.-T. Leung. The google file system. *SIGOPS Oper. Syst. Rev.*, 37(5):29–43, 2003.

[3] M. Lux and S. A. Chatzichristofis. Lire: lucene image retrieval: an extensible java cbir library. In *MM '08: Proceeding of the 16th ACM international conference on Multimedia*, pages 1085–1088, New York, NY, USA, 2008. ACM.

[4] T. White. *Hadoop The Definitive Guide.* O'Reilly Media, 2009.